VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Informační portál pro správu a vyhodnocení dotazníku pro firmu XO Studio

Information Portal for the Management and Evaluation of Questionnaires for the Company XO Studio

2013 Ivo Michalík

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Ivo Michalík**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Informační portál pro správu a vyhodnocení dotazníků pro firmu XO Studio
Information Portal for the Management and Evaluation of Questionnaires for the Company XO Studio

Zásady pro vypracování:

Cílem práce je vytvoření webového portálu pro správu a vyhodnocení dotazníků s využitím OOP programovacího jazyka PHP a Yii frameworku rozšířeného o aplikační vrstvu vytvořenou ve firmě XO Studio.

1. Seznámení se s tvorbou webových aplikací v jazyce PHP a ostatních vývojových prostředích.
2. Seznámení se s Yii framework pro tvorbu internetových aplikací.
3. Seznámení se s aplikační vrstvou firmy XO Studio.
4. Seznámení se s možnostmi dynamické tvorby dotazníků a se statickým vyhodnocením.
5. Vytvoření kompletní analýzy webového portálu pro správu a vyhodnocení dotazníků (specifikace požadavků, datová anlýza, funkční analýza, návrh implementace).
6. Praktická realizace webového portálu pro správu a vyhodnocení dotazníků (vytvořené pomocí Yii framework využívající aplikační vrstvu firmy XO Studio).
7. Nasazení systému do reálného provozu.

Seznam doporučené odborné literatury:

[1] GUTMANS, Andi; BAKKEN, Stig Sather; RETHANS, Derick. Mistrovství v PHP 5. Computer Press, 2007. 656 s. ISBN 978-80-251-1519-0, EAN: 9788025115190.
[2] CROFT, Jeff; LLOYD, Ian; RUBIN, Dan. Mistrovství v CSS : Pokročilé techniky pro webové designéry a vývojáře. Computer Press, 2007. 416 s. ISBN 978-80-251-1705-7, EAN: 9788025117057.
[3] HARWANI, B.M. JQuery Recipes : A Problem-Solution Approach. New York : Springer, 2010. 422 s. ISBN 978-1-4302-2709-0, ISBN-13 (electronic): 978-1-4302-2710-6.
[4] WINESETT, Jeffrey. Agile Web Application Development with Yii1.1. and PHP 5. 32 Lincoln Road Olton Birmingham, B27 6PA, UK. : Packt Publishing Ltd., 2010. 348 s. ISBN 978-1-847199-58-4.
[5] Skripta z předmětů – Teorie zpracování dat, Databázové a informační systémy, Úvod do softwarového inženýrství

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Štefan Bartoš**

Datum zadání: 18.11.2011

Datum odevzdání: 07.05.2013

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Souhlasím se zveřejněním této bakalářské práce dle požadavků čl. 26, odst. 9 *Studijního a zkušebního řádu pro studium v bakalářských programech VŠB-TU Ostrava.*

Požaduji, aby data, která jsou obsažena v bakalářské práci, nebudou dále zveřejňována, rozšiřována a jiným způsobem zneužita. Po obhajobě bakalářské práce žádám o znepřístupnění dat, popřípadě vymazání dat ze školního systému, pokud je to možné.

V Ostravě 29. dubna 2013 .....

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 29. dubna 2013 .....

Rád bych na tomto místě poděkoval Ing. Štefanu Bartošovi pracující ve firmě XO Studio za trpělivost, pomoc a čas se mnou strávený. Bez jeho poznatků a pomoci bych bakalářskou práci nezvládl.

Abstrakt

Bakalářská práce se zabývá moderním frameworkem Yii, který je určen pro jazyk PHP. Hlavní částí bakalářské práce je popis samotného frameworku, dále je to aplikační vrstva firmy XO Studio, která je postavena na tomto frameworku. Tento celek má za úkol usnadnit vývoj webových aplikací. V druhé části bakalářské práce je praktická realizace webového portálu ve frameworku Yii na vrstvě XO, který je určen pro správu dotazníků.

Klíčová slova: Framework, JSON, MVC, PHP, XO, Yii

Abstract

This thesis deals with modern framework Yii, which is designed for PHP. The main part of the thesis is the description of the framework, then it is an application layer from company XO Studio, which is based on this framework. This unit is aimed at facilitating the development of web applications. In the second part of the thesis there is the practical implementation of the web portal in the Yii framework based on layer XO, which is designed to manage questionnaires.

Keywords: Framework, JSON, MVC, PHP, XO, Yii

Seznam použitých zkratek a symbolů

CRUD	–	Create, Read. Update, Delete
DAO	–	Data Access Object
DFD	–	Data Flow Diagram
DRY	–	Don't Repeat Yourself
HTML	–	Hyper Text Markup Language
JSON	–	JavaScript Object Notation
ORM	–	Object-Relational mapping
PHP	–	Hypertext Preprocessor
SEO	–	Search Engine Optimization
SQL	–	Structured Query Language
URL	–	Uniform Resource Locator

Obsah

Seznam tabulek

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

V této bakalářské práci se budu zabývat webovým frameworkem Yii, který je napsaný v jazyce PHP, jeho možnostmi a architekturou. Dále do tohoto frameworku zakomponuji aplikační vrstvu firmy XO Studio (dále jen XO). Posléze se budu věnovat analýze webového portálu a jeho praktické realizaci.

V následující kapitole „Základní pojmy“ si objasníme problematiku webových frameworků, jejich architekturu a vzor MVC.

V kapitole 3 „Framework Yii“ je podrobněji rozebrán samotný framework, jeho struktura, použití a konfigurace. Dále nesmí chybět popis přístupu do databáze a konvence ulehčující práci s frameworkem a vývoj webových aplikací.

Ve 4. kapitole se rozepíšu o komponentách a modulech firmy XO Studio, na kterých jsem realizoval svůj projekt. Popíšu hlavní rysy této vrstvy, instalátor nových modulů, nesmí chybět internacionalizace a způsob vytváření a kontroly nad webovými aplikacemi.

Pro realizaci projektu je nutné vypracovat kompletní analýzu a návrh. Vše je zdokumentováno v 5. kapitole. Jsou sepsány specifikace požadavků, uživatelské role, které se systémem budou pracovat a k tomu standartní datová a funkční analýza.

V poslední 6. kapitole je stručně sepsáno, co jsem musel pro vytvoření aplikace podniknout, popřípadě jaké nastaly problémy. Rovněž je v této kapitole sepsán návod, jak si projekt nainstalovat a spustit. Případně je možnost zhlednout a vyzkoušet si celou aplikaci dostupnou na internetu.

2 Základní pojmy

2.1 Návrhové vzory, webové frameworky

2.1.0.1 Návrhový vzor Návrhový vzor představuje obecné řešení problému, který se opakovaně objevuje při návrhu softwaru. Návrhový vzor není knihovnou nebo částí zdrojového kódu, která by se dala přímo vložit do našeho programu. Jedná se o popis či šablonu, jak řešit problém způsobem, který může být použit v různých situacích. Objektově orientované návrhové vzory typicky ukazují vztahy a interakce mezi třídami a objekty, aniž by určovaly implementaci konkrétní třídy. Algoritmy nejsou považovány za návrhové vzory, protože řeší výpočetní problémy a nikoliv návrhové [1].

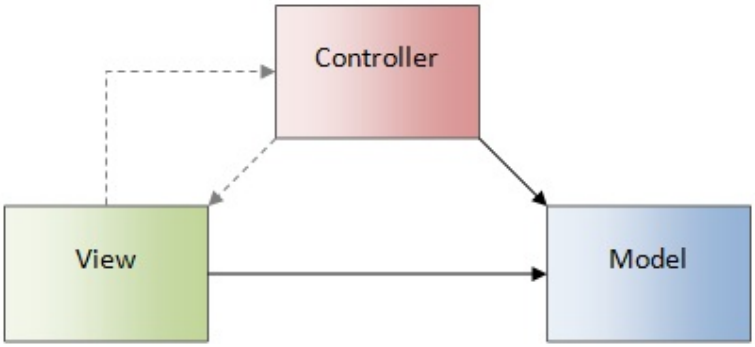
2.1.0.2 Webový framework Webové frameworky jsou softwarové nástroje, které se běžně používají při vytváření různých webových aplikací. Framework může zahrnovat vytváření a spouštění stránek pro webovou aplikaci nebo zajišťování a poskytování široké škály webových služeb. Framework zahrnuje všechny prvky potřebné k provedení požadovaných úkolů, čímž se eliminuje potřeba zajistit nezbytné nástroje z různých zdrojů. Jedním z klíčových prvků každého webového frameworku je softwarová knihovna. Jak už název napovídá, softwarové knihovny jsou centrální úložiště pro všechny typy aplikací, které mohou být využity při tvorbě a používání funkcí [2].

2.1.0.3 Využití frameworku Efektivní webový framework rovněž obsahuje základní funkce, které jsou nezbytné pro správu dat uložených v databázi, kde jsou webové stránky umístěny. Současně framework rovněž poskytuje možnost používat různé vzhledy pro stejnou stránku. Díky tomu je možné změnit pozadí na webových stránkách bez dopadu na uspořádání obrázků, textu a dalších prvků na stránkách a např. umožnit uživatelům si nastavit individuální vzhled pro stránky [2].

2.2 Návrhový vzor Model-view-controller

2.2.0.4 Vzor MVC Vzor Model-view-controller (MVC) dělí aplikaci na 3 logické části tak, aby je šlo upravovat samostatně a dopad změn byl na ostatní části co nejmenší. Tyto tři části jsou Model, View a Controller.

- **Model** - představuje tu část aplikace, která vykonává práci, pracuje s daty – řeší problém a poskytuje řešení. Tato a ani jiná část neřeší problém uchovávání dat (např. do databáze)
- **View** - (zobrazení) – zajišťuje prezentaci (zobrazení) dat koncovému uživateli, typicky uživatelské rozhraní
- **Controller** - zajišťuje řízení předchozích dvou částí, modelu a view. Reaguje na pokyny uživatele a rozhoduje jak se má změnit nebo chovat model a co se má zobrazit pomocí view



Obrázek 1: Schéma návrhového vzoru MVC

2.2.0.5 Vztahy v MVC Podstata MVC je jeho hlavní výhodou. Aplikace je rozdělena na tři části, každá část se zabývá určitým typem úkolů. Část model nemusí nic vědět o tom, jak data prezentovat uživateli. Obdobně část View pouze převezme výsledky od části Model a postará se o jejich zobrazení nebo předá Modelu příkaz od uživatele. Koordinaci těchto dvou částí zajišťuje Controller. Schéma návrhového vzoru MVC je zobrazeno na obrázku 1.

2.2.0.6 Popis schématu MVC Jak naznačují šipky, v principu existují pouze dvě přímé vazby, a to Controller má přímý odkaz na Model, aby mohl upravit jeho data a View má přímý odkaz na Model, aby mohl jeho data zobrazit. Žádné další vazby nejsou na této úrovni podstatné, ačkoliv v praxi a v závislosti na konkrétní variaci MVC je ještě poměrně častá vazba mezi Controllerem a View (někdy jednosměrná, někdy obousměrná) [3].

2.2.0.7 Vazby modelu Co naopak nikdy nesmí existovat je přímá vazba Modelu na ostatní dvě komponenty. V některých schématech uvidíte „nepřímou vazbu“ z Modelu na View, čímž se má na mysli, že pokud se data Modelu změní, příslušná View jsou upozorněna nějakým notifikačním mechanismem (např. pomocí vzoru Observer). V žádném případě však Model nemůže držet přímý odkaz na View nebo na Controller – to by byla hrubá chyba v návrhu aplikace [3].

2.3 JSON

2.3.0.8 Popis JSONu JSON je odlehčený formát pro výměnu dat. Je jednoduše čitelný a zapisovatelný. Je založen na podmnožině Programovacího jazyka JavaScript. JSON je textový, na jazyce zcela nezávislý formát, využívající zásady, dobře známé programátorům, jazyků rodiny C (C, C++, C#, Java, JavaScript, Perl, Python a dalších). Díky tomu je JSON pro výměnu dat opravdu ideálním jazykem. JSON je založen na dvou strukturách:

- Kolekce párů název/hodnota. Ta bývá v rozličných jazycích realizována jako objekt, záznam (record), struktura (struct), slovník (dictionary), hash tabulka, klíčový seznam (keyed list) nebo asociativní pole.
- Tříděný seznam hodnot. Ten je ve většině jazyků realizován jako pole, vektor, seznam (list) nebo posloupnost (sequence).

Jedná se o univerzální datové struktury a v podstatě všechny moderní programovací jazyky je v nějaké formě podporují. Je tedy logické, aby na nich byl založen i nezávislý výměnný formát [4].

2.3.1 Struktura JSONu

V JSON jsou realizovány tyto struktury s využitím následujících konstrukcí:

- *Objekt* - je netříděná množina párů název/hodnota. Je uvozen znakem { (levá složená závorka) a zakončen znakem } (pravá složená závorka). Každý název je následován znakem : (dvojtečka) a páry název/hodnota jsou pak odděleny znakem , (čárka).
- *Pole* - je setříděnou kolekcí hodnot. Začíná znakem [(levá hranatá závorka) a končí znakem] (pravá hranatá závorka). Hodnoty jsou odděleny znakem , (čárka).
- *Hodnota* - rozumíme tím řetezec uzavřený do dvojitých uvozovek, číslo, true či false, null, objekt nebo pole. Tyto struktury mohou být vnořovány.
- *Řetězec* - je nula nebo více znaků kódování Unicode, uzavřených do dvojitých uvozovek a využívající únikových sekvencí (escape sequence) s použitím zpětného lomítka. Znak je reprezentován jako řetězec s jediným znakem. Řetězec je velmi podobný řetězcům z jazyků C nebo Java.
- *Číslo* - je podobné číslům z jazyků C a Java. Jedinou výjimkou je, že není používán oktalový ani hexadecimální zápis [4].

3 Framework Yii

3.0.1.1 Úvod Yii je vysoce výkoným komponentově orientovaným PHP frameworkem, vynikající pro vývoj rozsáhlých webových aplikací. Umožňuje maximální znovupoužitelnost v programování webových stránek a může výrazně zrychlit Váš proces vývoje webových aplikací. Název Yii (vyslovuje se Yee nebo [:ji]) je zkratkou pro "Yes It Is!"(v překladu "Ano, je!"). Je to často přesná a nejvýstižnější odpověď na dotazy od těch, co nejsou obeznámeni s frameworkem např.:

Je to rychlé? ... Je to bezpečné? ... Je to profesionální? ... Je to to pravé pro můj další projekt? ... Ano, je! [5]

3.1 Efektivita a rozšiřitelnost

3.1.0.2 Přehledný kód s pomocí MVC Yii je navržen tak, aby Vám pomohl s vývojem DRY. DRY (Neopakuj sám sebe) je klíčový pojem pro vývoj agilních aplikací. Všechny aplikace napsané v Yii jsou postaveny s využitím architektury MVC, kterou jsme si již popsali výše. Yii vynucuje tento vzor tím, že udržuje každou část MVC kódu odděleně. Tím minimalizuje duplicitu a pomáhá podporovat opětovné používání kódu a snadnou jeho udržovatelnost. Čím méně kódu musíte napsat, tím rychleji se Vaše aplikace dostane na trh. Neboli, čím snazší je provádět údržbu své aplikace, tím déle zůstane na trhu [5].

3.1.0.3 Připraven na rozšiřování Yii byl navržen tak, aby téměř každý kousek svého kódu mohl být rozšířen či přizpůsoben téměř jakýmkoliv potřebám nebo požadavkům. Pokud chcete, aby se vaše rozšíření stalo užitečným nástrojem pro ostatní vývojáře, poskytuje Yii různé tutoriály, které vám pomohou vytvořit rozšíření třetích stran. Yii je nabitý funkcemi, které vám pomohou splnit vysoké nároky kladené na dnešní webové aplikace. Ajaxové widgety, integrace webových služeb, prosazování MVC architektury, DAO a relační Active Record databázová vrstva, sofistikovaný caching, hierarchické role založené na řízení přístupů, motivů, internacionalizace (I18N) a lokalizace (L10N), to vše je jen špičkou ledovce [5].

3.2 Struktura frameworku

Jak již bylo zmíněno dříve, Yii používá vzor MVC a poskytuje explicitní strukturu složek pro každou část MVC kódu. Pojďme se podívat, jak Yii zavádí a prosazuje MVC architekturu.

3.2.1 Model

3.2.1.1 Dva typy modelu Model v Yii je instancí třídy `CModel` nebo jejím potomkem. Třída model typicky zahrnuje datové atributy, které mohou mít samostatné štítky (název či krátký popis atributu pro pochopení jeho významu) a můžou být ověřeny pomocí pravidel, které jsou definované v modelu. Údaje, které tvoří atributy v modelu, můžou

pocházet z řádku tabulky v databázi nebo ze vstupních polí umístěných ve formulářích. Yii implementuje dva druhy modelů: model formuláře (třída `CFormModel`) a model active record (aktivní záznam) (třída `CActiveRecord`). Oba dva rozšiřují stejnou základní třídu `CModel`.

3.2.1.2 Form Model Třída `CFormModel` reprezentuje datový model, který shromažďuje vstupy z formulářů a zapouzdřuje celou logiku pro validaci polí formuláře. Po vytvoření a validaci si tyto údaje uloží do paměti nebo s pomocí modelu AR (active record) může data uložit do databáze.

3.2.1.3 Active Record Model Active Record (AR) neboli Aktivní záznam je návrhový vzor pro přístup do databáze v objektově orientovaném prostředí. Každý AR objekt je instancí třídy `CActiveRecord` nebo jejího potomka, která obtéká jeden řádek v tabulce nebo pohledu a zapouzdřuje veškerou logiku a detaily týkající se přístupu do databáze. Datové hodnoty polí pro každý sloupec v řádku tabulky jsou reprezentovány jako vlastnosti objektu AR.

3.2.2 View

View v Yii je PHP skript, který obsahuje prvky související s uživatelským rozhraním, často postavené pomocí HTML, ale také může obsahovat PHP příkazy. PHP příkazy uvnitř View jsou velmi jednoduché podmínky nebo smyčky a nebo odkazují na jiné související prvky, jako jsou pomocné HTML metody tříd nebo předkompilované widgety. Více sofistikovanější logika by měla být oddělena od View a umístěna vhodně buď v modelu (je-li v přímém styku s daty) nebo v controlleru.

3.2.3 Controller

Controller je naším hlavním manažerem požadavků a je odpovědný za přijetí vstupů od uživatele, interakci s modelem a konstruování View tak, aby bylo zobrazeno správně. Controler v Yii je instancí třídy `CController` nebo jejího potomka. Když je Controller spuštěn, provede požadovanou akci, která pak komunikuje s potřebnými modely a poté vykreslí odpovídající View. Akce, ve své nejjednodušší podobě, je metodou třídy Controlleru, jejíž název začíná slovem "action".

3.3 Spuštění frameworku

Pro nainstalování a spuštění frameworku je potřeba webový server s podporou PHP verze 5.1.0 nebo vyšší. Framework je dostupný ke stažení na této adrese:
`http://www.yiiframework.com/download/`. Po rozbalení souboru se musí do výchozího adresáře serveru zkopírovat složka "framework"a tím je framework připraven k použití. Vytvoření nové aplikace se provádí přes konzoli. Po spuštění příkazového řádku se přepneme do složky frameworku a poté nad souborem `yiic` spustíme jednoduchý příkaz "webapp", který za nás vytvoří kompletní strukturu základní aplikace.

```
$ cd /var/www/framework
$ ./yiic webapp ../demoaplikace
```

3.3.1 Struktura aplikace

Takto vypadá základní struktura aplikace po použití příkazu:

```
framework
demoaplikace
    assets
    css
    images
    protected
        components
        config
        controllers
        extensions
        messages
        models
        modules
        views
        .htaccess
        yiic.bat
        yiic.php
    themes
    index.php
    .htaccess
```

3.3.1.1 Stručný popis adresářů Do adresáře `assets` se ukládají veškeré soubory, které neobsahují HTML či PHP kód. V podstatě jsou tam ukládány šablony css, js (javascript) soubory, obrázky a další, ze všech složek a podsložek, aby byly přístupné všem případným modulům a nevznikly zde konflikty s pojmenováním souborů. Yii navíc poskytuje kompletního manažera v podobě komponenty `CAssetManager`. Do složky `css` se ukládají šablonové css soubory, ve složce `images` jsou pak uloženy obrázky. Je zde ještě složka themes, do které můžeme ukládat různé layouty (rozložení stránky).

3.3.1.2 Adresář protected Toto je nejdůležitější složka, která vytváří funkčnost celé aplikace. V ní se nachází složky:

- `components` - Zde se ukládají pomocné komponenty, které slouží např. jako základ modelu, controlleru nebo jiných tříd určené k budoucímu využití.

- `config` - Zde se nachází soubor `main.php`, ve kterém je uložena konfigurace naší aplikace.
- `controllers` - Tady jsou uloženy controllery. Výchozím controllerem je `SiteController`.
- `extensions` - Zde se ukládají různé doplňky třetích stran (widgety).
- `messages` - Do této složky se ukládají skripty používané pro překlad textu a hlášek do jiných jazyků než je výchozí.
- `models` - Tato složka slouží k ukládání modelů.
- `modules` - Zde jsou ukládány moduly. `Modul` můžeme chápat jako menší celek, který se rovněž skládá z controlleru, modelů a view.
- `views` - Jak název napovídá, zde jsou uloženy view, nejprve složka s názvem ID Controlleru např. `clanek` a v ní jsou umístěny pak jednotlivé view jako `index.php` či `detail.php`.

3.4 Používání frameworku

Yii přichází se dvěma způsoby, jak si můžete od frameworku vyžádat automatické generování kódu. Tím mám na mysli, že pokud máme aplikaci vytvořenou a v konfiguraci nastavenou databázi, může si nechat automaticky vygenerovat kód pro modely, controllery či CRUD operace. CRUD operace slouží k práci s daty (vytvořit, číst, změnit, smazat).

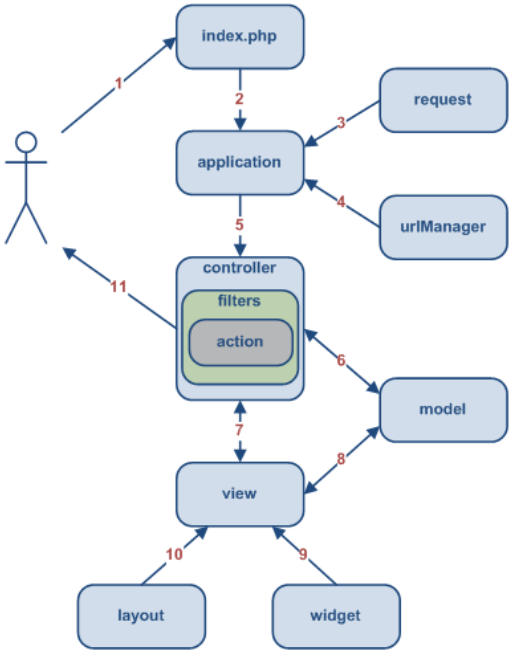
3.4.0.3 Příkazový řádek Pomocí konzole máme možnost si nechat kód vygenerovat. Stačí nad souborem `yiic` zadat příkaz `shell`. Ten obsahuje další sadu příkazů, pomocí kterých dokážeme vygenerovat kód pro jednotlivé části naší aplikace.

3.4.0.4 Grafický nástroj Yii přichází dokonce s webovým rozhraním "Gii", což je webová kopie interaktivního shellu. Stačí jenom odpoznámkovat příslušnou sekci v konfiguračním souboru, nastavit si heslo a zadat adresu do prohlížeče ve tvaru `http://WebRoot/index.php?r=gii`.

3.5 Průběh aplikace

Následující diagram na obrázku 2 ukazuje typický průběh Yii aplikace, když je zadán požadavek od uživatele:

1. Uživatel zadá požadavek napsáním URL adresy do prohlížeče např.: `http://www.example.com/index.php?r=post/show&id=1` a webový server zpracuje žádost spuštěním zaváděcího skriptu `index.php`.



Obrázek 2: Typický průběh požadavku v Yii

2. Zaváděcí skript vytvoří instanci třídy `Application` a spustí ji.
3. Aplikace získá informace o požadavku uživatele z aplikační komponenty `request`.
4. Aplikace stanoví požadovaný Controller a akci s pomocí aplikační komponenty `urlManager`. V našem případě, controller je `post` (část adresy, kde je `post/show`), který odkazuje na třídu `PostController` a akce je `show`.
5. Aplikace vytvoří instanci požadovaného controlleru. Controller zjistí, že akce `show` odkazuje na metodu pojmenovanou `actionShow` ve třídě controlleru. Pokud akce projde všemi filtry a dostane povolení, je spuštěna.
6. Akce načte model `Post` z databáze, jehož ID je 1.
7. Poté akce načte view pojmenované `show` společně s modelem `Post`.
8. View si přečte a zobrazí atributy modelu `Post`.
9. View spustí doprovodné widgety (doplňky), pokud jsou potřeba.
10. Výsledek view je zabalen do vzhledu stránek.
11. Akce dokončí vykreslování view a zobrazí výsledek uživateli.[6]

3.6 ORM a Active Record

3.6.0.5 ORM Webové aplikace udržují svá data obvykle v relační databázi. Nicméně, aplikace potřebují tyto data z databáze namapovat do paměti. Knihovny ORM (Objektově-relační mapování) poskytují mapování databázových tabulek třídám doménových objektů. Většina kódu, která se zabývá ORM, popisuje, jak jednotlivá pole v databázi odpovídají polím v našich objektech. Naštěstí, Yii nám přichází na pomoc tím, že ORM vrstva je napsaná ve formě vzoru AR (Active Record).

3.6.0.6 Mapování v AR Jak již bylo dříve zmíněno, AR je návrhový vzor pro přístup do databáze v objektově orientovaném prostředí. Mapuje tabulky na třídy, řádky na objekty a sloupce na parametry objektů. Jinými slovy, každá instance třídy `CActiveRecord` nebo jejího potomka představuje jeden řádek v databázové tabulce. Navíc třída AR poskytuje kompletní logiku, která je uplatňována na tato data. V konečném výsledku je to třída, která definuje vše o tom, jak by se mělo zapisovat a číst z databáze.

3.6.0.7 Jednoduché používání AR Implementace AR v Yii šetří spoustu času, obvykle stráveného psaním nudných a opakujících se SQL příkazů, které jsou potřeba pro čtení a zápis dat z databáze. Navíc umožňuje vývojařům přistupovat k datům mnohem více objektově orientovaným způsobem. Pro představu, ve výpisu 1 je příklad kódu, který ukazuje, jak jednoduše se s AR pracuje. Nejprve si načteme model z databáze, který má ID 99. Poté změníme název a následně uložíme veškeré změny.

```
$post = Post::model()->findByPk(99);
$post->title ="nazev";
$post->save();
```

Výpis 1: Ukázka práce s modelem pomocí AR

3.7 Konvence

Yii upřednostňuje konvenci před konfigurací. Používáním určitých zásad je možné vytvářet sofistikované aplikace bez psaní složitých konfigurací. Nyní si popíšeme, které zásady jsou doporučeny pro programování v Yii. Předpokládejme, že `WebRoot` je složka, kde je aplikace nainstalována.

3.7.1 URL

Ve výchozím nastavení rozpoznává Yii URL adresu v následujícím formátu:
`http://WebRoot/index.php?r=ControllerID/ActionID`
Proměnná `r` je pomocí Yii rozdělena na controller a akci. Pokud je vynechána `ActionID`, použije controller výchozí akci. A pokud je vynechán i `ControllerID` (proměnná r je prázdná), aplikace použije výchozí controller. Navíc s pomocí komponenty `CUrlManager`

je možné vytvářet a rozpoznávat jednodušší tzv. SEO adresy např.
`http://WebRoot/ControllerID/ActionID.html`.

3.7.2 Kód

3.7.2.1 Základní pojmenování Yii doporučuje pojmenování proměnných, funkcí a názvů tříd tak, že každé nové slovo v názvu napíšeme velkým písmenem a spojíme bez mezer. Navíc proměnné a funkce by měli mít první písmeno napsané malým, abychom jej lépe rozeznali od názvů tříd (např. `$povinnaPolozka`, `zkontrolujNazvy()`, `ClanekModel`). Dále je doporučeno, aby se třídy pojmenovávaly nějakým unikátním způsobem, aby nedocházelo ke konfliktu názvů tříd třetích stran. Z tohoto důvodu začínají všechny třídy frameworku Yii velkým písmenem "C".

3.7.2.2 Speciální pravidlo Speciální pravidlo pro pojmenování controllerů je to, že za Váš název ještě připojíte slovo `Controller` bez mezery. Controller ID je pak definován tak, že se slovo `Controller` z názvu vyhodí a ze zbytku názvu se první písmeno převede na malé. Například třída `ClanekController` bude mít ID `clanek`. Toto pravidlo zajišťuje aplikaci větší zabezpečení. Navíc použitím tohoto pravidla jsou i adresy o něco čistší (např. `/index.php?r=clanek/index` namísto
`/index.php?r=ClanekController/index`).

3.7.3 Konfigurace

Konfigurace v Yii je tvořena polem, které je tvořeno pomocí párů key-value (klíč-hodnota). Každý klíč představuje název vlastnosti objektu. Například `array('name' => 'Moje aplikace')` nám nastaví název aplikace na "Moje aplikace". Pokud nějaká vlastnost objektu není nastavena v konfiguraci, Yii ji přiřadí její výchozí hodnotu.

3.7.4 Soubory

Zásady pro pojmenování souborů závisí na jejich typu. Soubory s třídami by měly být pojmenovány podle třídy, kterou obsahují. Například třída `CController` je umístěna v souboru `CController.php`. Každý soubor s třídami by měl obsahovat jednu třídu pro lepší přehlednost. Soubory s view by měly být pojmenovány podle názvu view. Například `index` view je umístěn v souboru `index.php`. Soubor view je PHP skript, který obsahuje HTML a PHP kód určen především pro prezentační účely. Konfigurační soubory lze pojmenovat libovolně. Konfigurační soubor je totiž PHP skript, jehož jediným cílem je vrátit pole představující nějakou konfiguraci.

3.7.5 Databáze

Většina webových aplikací používají nějakou databázi. Pro běžnou praxi se doporučuje používání několika základních zásad:

- Názvy tabulek a sloupců jsou psány malými písmeny.

- Slova v názvech oddělujeme pomocí podtržítka (např. `zbozi_v_objednavce`)
- Pro názvy tabulek používáme buď jednotné číslo nebo množné číslo. Nikdy ne obojí najednou. Pro jednoduchost se doporučuje používání jednotného čísla.
- Názvy tabulek rovněž mohou obsahovat předponu (např. `tbl_`). To je obvzlášť výhodné, pokud tabulky jedné aplikace jsou uloženy v databázi společně s tabulkami jiné aplikace. Poté můžeme jednoduše rozlišit, která tabulka patří ke které aplikaci.

4 Aplikační vrstva XO Studio

V této kapitole se Vám budu snažit popsat aplikační vrstvu od firmy XO Studio (dále jen XO). Začnu nejprve se zásadami, které se ve firmě dodržují.

4.1 Konvence

4.1.0.1 Struktura aplikace Základní adresář tvoří pouze dvě složky a to `protected`, která již byla zmíněna ve 3. kapitole a složka `writable` (v překladu zapisovatelný). Do této složky se ukládají veškeré assety, nahráváné soubory, obrázky a v poslední řadě rovněž log (soubory se záznamy průběhu aplikace). Mně osobně se velice líbí, že veškerý obsah dostupný návštěvníkům, je umístěn v jedné složce na jednom místě. Nyní se dostávám ke složce `protected`. V ní najdeme poněkud méně složek, než když nám framework vytvoří základní aplikaci.

- `components` - Jak již bylo zmíněno výše, jsou zde uloženy veškeré komponenty. Zde je umístěna většina práce firmy a to konkrétně všechny komponenty s názvem začínající XO.
- `config` - Zde je to beze změny a nachází se zde soubor `main.php` s konfigurací aplikace.
- `extensions` - Zde jsou uloženy widgety (doplňky). Od XO zde máme XOTree. Což je doplňek, který slouží k výpisu dat (např. Kategorie) pěkně ve stromové struktuře.
- `modules` - Zde jsou uloženy dva moduly a to `admin` a `frontend`. Každý modul představuje jeden celek aplikace a jak již správně tušíte, část `admin` zpracovává kompletní administraci webových stránek, kdežto modul `frontend` (popředí) má na starost prezentaci stránek, které jsou zobrazovány návštěvníkům. Tyto dvě části popíšu později.

4.1.0.2 Databáze Zde jsou uplatněny zásady doporučené frameworkem Yii, které již byly zmíněny výše. Databáze je typu MySQL.[1] XO zavádí další pravidla pro přehlednější a jednodušší práci s databází.

- Všechny tabulky jsou uloženy ve stejném formátu, konkrétně MyISAM (formát úložiště dat).
- Veškeré názvy sloupců a tabulek jsou pojmenovány anglicky v jednotném čísle, komentáře jsou psány česky.
- Všechny tabulky používají jednu ze tří předpon: `code_`, `rel_` a `tab_`. Předpona `code_` označuje všechny tabulky, které se nazývají tzv. číselníky (tabulky, které neobsahují žádné cizí klíče a poskytují data jiným tabulkám).

[1] Yii samozřejmě umožňuje používání i jiných typů databáze např. MSSQL, SQLite a další.

Dále předpona `rel_` slouží k označení všech vazebních tabulek (pomocné tabulky, kombinující dvě jiné tabulky). Poslední předpona `tab_` označuje všechny ostatní tabulky.

- Většina tabulek má i svoji textovou tabulku pojmenovanou úplně stejně, akorát je na konci přidáno `_text`. V této tabulce jsou uloženy veškeré údaje, které jsou textového charakteru a slouží k internacionalizaci (popsáno později). Pokud máme např. tabulku `tab_clanek`, kde máme uloženo ID článku, uživatele a datum, pak v druhé tabulce `tab_clanek_text` máme uloženo ID z první tabulky, ID jazyka, titulek a obsah článku.
- Všechny sloupce a tabulky také mají přiřazen krátký komentář, který vystihuje, k čemu je daná tabulka či sloupec používán. Komentář u tabulek mívá popisek např. "Modul Blog"a u textové tabulky "Textová část modulu Blog".

4.2 Instalace modulů

4.2.0.3 Použití instalátoru Firma má připravený i instalátor pro zavádění nových modulů. Po vytvoření základní struktury nového modulu, zavoláme adresu instalátoru:
`http://WebRoot/instalace/nazev_noveho_modulu`
Instalátor vytvoří příslušné tabulky v databázi a rovněž přidá záznamy do tabulek, ve kterých jsou uložena práva pro uživatele. Samozřejmě instalátor se vůbec nemusí používat, pokud je člověk zdatný databázista, dokáže SQL příkazy spustit sám. Ale je to příjemná pomůcka, která nám dokáže ušetřit čas.

4.3 Internacionalizace

Pod tímto dlouhým a cizím slovem se skrývá multijazyčnost aneb možnost si zobrazit stejné webové stránky ve více jazycích. XO zde přichází s komplexním řešením tohoto problému.

4.3.0.4 Databáze V podkapitole Konvence byla řeč o tom, že tabulky, jenž obsahují data textového charakteru, jsou obsaženy v jiné (nazývané textové) tabulce. Všechny textové tabulky obsahují tyto sloupce:

- id
- id_nazev-modulu - id, které odkazuje na id převzaté z netextové tabulky
- id_language - id jazyka, ve kterém jsou data uložena
- title - titulek neboli název daného objektu
- seo_url - URL adresa, která je tvořena převedeným titulkem bez diakritiky a místo mezer jsou pomlčky
- seo_keyword - zde mohou být uložena klíčová slova popisující objekt

- seo_description - objekt rovněž může mít uložen krátký popis o čem je jeho obsah nebo jeho využití

Nyní pro ilustraci uvedu tabulku `tab_clanek`, která uchovává data o článcích. Pro ukázku tabulka obsahuje sloupce jako id (ID), id_user (id uživatele, který článek napsal), id_kategroie (článek je v nějaké kategorii). Naše textová tabulka `tab_clanek_text`, kromě výše zmíněných atributů, rovněž obsahuje atribut content (obsah či chcete-li samotný článek). Díky tomu můžeme článek napsat např. česky, poté třeba německy. Při uložení článku se do tabulky `tab_clanek` přidá jeden záznam o článku a do textové tabulky `tab_clanek_text` se přidají dva záznamy. Jeden bude obsahovat údaje v češtině, druhý v němčině.

4.3.0.5 Skript pro výpis Pro výpis daného modelu v nastaveném jazyce slouží komponenta `XOInterModel`. Tato třída dědí z třídy `CActiveRecord` a doplňuje ji o metody, které nám slouží k internacionalizaci. Jedna z nejdůležitějších a nejpoužívanějších metod je metoda `getInternationalModel`. Tato metoda je zavolána na náš základní model. Metoda si sama zjistí o jakou třídu se jedná a s pomocí parametru, kde je nastaveno ID jazyka, vytvoří nový textový model a vrátí ho. V ukázce 2 je krátký skript, jak se metoda používá.

```
$clanek = Clanek::model()->findByPk(11);
$clanektext = $clanek->getInternationalModel(1);
echo $clanektext->content;
```

Výpis 2: Ukázka načtení internacionalizačního modelu

Nejprve si článek s ID 11 načteme z databáze. Poté si vyžádáme textový model pomocí zmíněné metody, kde 1 znamená ID jazyka, ve kterém chceme náš článek vypsat. Nakonec vytiskneme obsah samotného článku.

4.4 Část Frontend

Tato část se stará o zobrazení webových stránek všem návštěvníkům.

4.4.0.6 Struktura Struktura je úplně stejná, jako když framework vytváří novou aplikaci. Čili zde najdete složky jako assets, components, controllers, models, modules a views, jejichž význam jsme si již vysvětlili. Důležité je to, že ve složce views je složka layouts, která se stará o to, jakou strukturu mají jednotlivé stránky mít. Layout je poskládán ze tří souborů, a to header.php (hlavička stránky a vše, co je umístěné nad hlavním obsahem), menu.php (pokud máme navigační menu, tak ho vypíšeme) a footer.php (patička stránky a vše, co je umístěné pod obsahem).

4.4.0.7 Moduly Ve složce modules jsou uloženy jednotlivé moduly. Všechny názvy modulů začínají malým písmenem "f"a poté následuje pravidlo pojmenovávání tříd. Ka-

ždý modul má na starost výpis jednotlivých částí stránky. Zde je krátký seznam modulů, který obsahuje všechny potřebné moduly pro zobrazení úvodní stránky:

- fFrontendMenu - tento modul se stará o výpis navigačního menu stránek (je volán v souboru menu.php)
- fPage - vypisuje nám hlavní obsah stránky textového charakteru
- fArticle - modul vypisující různé bloky umístěné na stránce (např. levý boční panel, reklama pod obsahem a další ...)

4.5 Část Admin

Administrační část poskytuje registrovaným uživatelům přidávat či měnit obsah stránek.

4.5.0.8 Struktura Struktura je úplně stejná jako u části frontend.

4.5.0.9 Moduly Názvy modulů však nezačínají malým písmenem "f". Nyní uvedu krátký seznam modulů, bez kterých administrace je nefunkční.

- core - hlavní modul, kontrolující, zda-li byl uživatel přihlášen a ověřen. Dále tento modul poskytuje hlavní nastavení aplikace, jazyků, jednotlivá práva uživatelů či přizpůsobení samotné administrační části.
- install - velmi malý a užitečný modul, který má na starost instalaci nových modulů.
- rights - další důležitý modul, který při každé akci kontroluje, zda-li dotyčný uživatel má právo na požadovanou akci. Pokud nemá, vypíše hlášku o nepovolených právech daného uživatele.
- user - modul udržující seznam všech uživatelů včetně administrátorů

Samozřejmě jsou zde rovněž další moduly, každý sloužící k nastavení obsahu modulu ve frontendu. Např. pro modul fArticle je v administrační části modul article a podobně.

4.6 Další doprovodné komponenty XO

4.6.0.10 Asset Manager Komponenta `XOAssetManager` se stará o publikování všech assets do složky `/writable/assets/`. Pokud je spuštěn jakýkoliv modul, postará se o publikování všech assets s modulem spojené.

4.6.0.11 URL Manager Tato velice komplexní a složitá komponenta se stará o to, aby uživatelé mohli používat více přívětivé SEO adresy. Adresa zadaná uživatalem je převedena do podoby, aby framework mohl zavolat příslušný controller a akci. Například uživatel zavolá adresu: `http://WebRoot/page/jak-spravne-zalozit-ucet/` Manager z adresy zjistí, že slovo `page` se vztahuje k modulu `fPage`, a proto zavolá příslušný controller. Nyní ještě potřebuje zjistit, jakou akci má spustit. Pokud by za `page/` následovalo slovo `vypis`, je spuštěna výchozí akce `vypis` či chcete-li `index`. V našem případě je spuštěna akce `detail`, která si zjistí ID tohoto objektu, načte data z databáze a nechá vykreslit požadavek.

4.6.0.12 XOListModel Tato komponenta je používána v administrační části. Je to třída, která dědí z třídy `XOInterModel`, která byla popsána výše. Tato třída navíc přidává pomocné metody pro jednodušší a pohodlnější práci v administračním prostředí. Zmíním aspoň dvě metody. Metoda `isAuthorized()` kontroluje, zda přihlášený uživatel má právo na požadovanou akci s modelem. Další metoda `loadTableJSONData()` slouží k výpisu seznamu objektu modelu pomocí JSON (princip fungování vysvětlen ve 2. kapitole). Po změně nastavení stránkování nebo vyhledávání je seznam akutalizován, aniž bychom museli obnovit stránku.

5 Návrh webové aplikace

5.1 Specifikace požadavků

Pro realizaci webové aplikace ve frameworku Yii byl zadán webový portál pro správu dotazníků. Jedná se o systém pro zadávání, vyplňování a vyhodnocení dotazníků.

5.1.0.13 Popis Potřebujeme evidovat dotazníky a veškeré odpovědi respondentů. Dotazník je dán svým názvem, krátkým popisem proč byl dotazník vytvořen či využití výsledků, počáteční a koncové datum ohraničující dobu zadávaní odpovědí a v poslední řadě, zda-li mají být výsledky veřejné či nikoliv. Dále potřebujeme evidovat otázky obsažené v dotazníku. Otázka je určena svým samotným zněním, zda-li je otázka povinná či nikoliv a také, jakého typu je daná otázka (např. jedna volba, textová odpověď atd.). Posledním významným objektem jsou samotné odpovědi respondentů, kde evidujeme kdy byl dotazník vyplněn a uložení hodnot samotných odpovědí na otázky.

5.1.0.14 Uživatelské role S webovým portálem budou moci pracovat 3 uživatelské role. První se nazývá "host"nebo-li návštěvník webových stránek. Ten si může prohlížet seznam dostupných dotazníků, vyplňovat je a případně zhlédnout výsledky dotazníků. Rovněž se bude moct registrovat a získat tak práva další role "user"(registrovaný uživatel). Tato role dokáže vše co předešlá role a zároveň bude moct spravovat svoje dotazníky, vytvářet nové či prohlížet si výsledky. Poslední rolí je "admin". Administrátoři budou mít kontrolu nad celým portálem.

5.1.0.15 Případy užití Nyní vypíšu funkce pro každou roli. Nejprve funkce dostupné roli "host":

- Výpis dotazníků - vypíše se seznam všech dostupných dotazníků, kterým ještě neskončilo datum platnosti
- Vyplnění dotazníku - kdokoliv může vyplnit dotazník. Navíc zde neplatí restrikce, že z jedné IP adresy jde dotazník vyplnit pouze jednou a tak z jednoho PC může vyplňovat vícero lidí
- Výpis výsledků (veřejné) - pokud má dotazník povolené veřejné výsledky, je možnost si zobrazit, jak byl dotazník vyhodnocen
- Registrace

Funkce dostupné roli "user":

- Přihlášení do systému
- Správa dotazníků - uživatel má práva na vytváření, editaci a mazání svých dotazníků

- Správa otázek a odpovědí - každému dotazníku je potřeba vytvořit otázky a pokud je nutné tak i volby odpovědí
- Výpis výsledků - každý uživatel si samozřejmě může nechat zobrazit výsledky všech svých dotazníků. Navíc má tu výhodu, že si výsledky může zobrazit, i když vyplňování dotazníku ještě neskončilo
- Změna údajů - možnost si změnit jméno, příjmení či emailovou adresu

Funkce dostupné roli "admin":

- Správa uživatelů - možnost změny či smazání uživatele
- Správa aplikace - dále je tu možnost základní nastavení aplikace (např. jazyky, titulky na stránkách a další ...)

5.2 Datová analýza

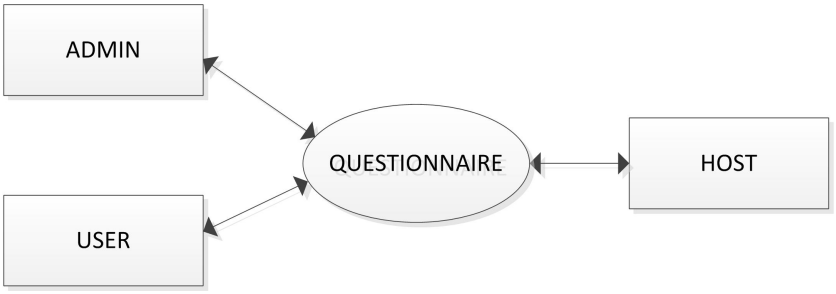
5.2.1 Návrh databázového modelu

V příloze je na obrázku 14 znázorněn databázový model. Následuje krátký komentář ke každé tabulce.

5.2.1.1 tab_user Podobná tabulka je umístěna ve většině frameworcích, a proto zde rovněž nemůže chybět. Využívám ji k identifikaci, který uživatel založil který dotazník. Obsahuje atributy: `id`->id uživatele, `id_language`->id jazyka z tabulky `code_language` pro nastavení jazyka v administračním prostředí, `login`->unikátní přihlašovací jméno uživatele, `password`->heslo v zašifrované podobě.

5.2.1.2 survey Tato tabulka eviduje jednotlivé dotazníky. `Title` je jednoznačný název průzkumu např. „Rekreační sport v zimě". Dále `description` obsahuje krátký popis, co zadavatel od dotazníku očekává a proč ho vytvořil. Atributy `start_date` a `expiration_date` určují OD kdy DO kdy je možné dotazník vyplňovat. Atribut `id_user` je cizí klíč a identifikuje, který uživatel daný průzkum založil. Poslední atribut `is_public` určuje, zda-li zadavatel chce, aby byly výsledky dotazníku zobrazeny široké veřejnosti.

5.2.1.3 question V této tabulce jsou uloženy otázky pro daný dotazník. Atribut `title` obsahuje celé znění otázky. Atr. `description` může obsahovat doporučení, jak na otázku odpovědět (např. co je myšlenkou otázky). Za další `answer_required` určuje, zda je otázka povinná či ne. `Position` určuje pořadí otázek v dotazníku. Dále máme cizí klíče, `id_survey` přiřazuje otázku k dotazníku, `id_question_type` určuje jakého typu je otázka.



Obrázek 3: Kontextový diagram

5.2.1.4 question_type Tento číselník obsahuje jednotlivé typy otázek (např. textová odpověď, volba jedné možnosti, více možností atd.). Atribut `code_type` obsahuje hodnoty např. "yesno", "singlechoice", "multichoice"a slouží ke stavbě otázky. Atribut `title` je pak samotný název zobrazen obyčejnému uživateli v seznamu.

5.2.1.5 choice Tato tabulka eviduje všechny volby odpovědí pro danou otázku. Má svoje `id`, dále samotná odpověď je uložena v atributu `value` a nechybí ani atribut `position` pro pořadí zobrazení odpovědí.

5.2.1.6 response Zde se evidují jednotliví respondenti co zodpovězeli dotazník, určeni jsou přes `id_response`. Dále jsou zde již informační atributy jako `date`, kdy respondent odeslal odpovědi a atribut `time_filling`, což je doba, po kterou respondent odpovídal na otázky.

5.2.1.7 Vazební tabulky mezi `response` a `question` Tabulky `text_response`, `number_response`, `yesno_response`, `choice_response` jsou vazební tabulky mezi otázkami a odpověďmi, protože na jednu otázku máte více odpovědí stejně tak jako jeden respondent odpovídá na více otázek. Atribut `value` nabývá pokaždé jiného datového typu podle toho, jaký je typ otázky. V tabulce `choice_response` není `value`, ale `id_choice`, který identifikuje odpověď z tabulky `choice`.

5.2.2 Popis atributů

V příloze je uvedena tabulka 1, která podrobněji popisuje atributy všech tabulek v databázi, jejich datové typy, velikost a další jejich základní vlastnosti.

5.3 Funkční analýza

5.3.0.1 DFD diagramy Na obrázku 3 je znázorněn kontextový diagram zobrazující role vstupující do systému. V příloze je na obrázku 9 zobrazen diagram 0 úrovně. Pro ilustraci následují čtyři další diagramy 1 úrovně, které mají za úkol přiblížit jakým způsobem

budou probíhat hlavní funkce systému. V následujícím odstavci jsou velice krátce popsány jednotlivé funkce.

5.3.0.2 Minispecifikace Zde jsou vypsány základní funkce pro správnou funkčnost systému, které jsou seřazeny podle úrovně a očíslování.

1. Správa os. profilu

   1.1 Změna údajů

   - Vstup: načtení `id_user` právě přihlášeného uživatele
   - Akce: po zavolání požadavku se předá `id_user` uživatele do SQL dotazu a načtou se data do modelu, poté se vykreslí formulář s daty z modelu. Po odeslání formuláře jsou data zvalidována a v případě úspěchu uložena do databáze.
   - Výstup: uložení dat do tabulky `tab_user_data`

   1.2 Zobrazení údajů

   - Vstup: načtení dat z tabulky `tab_user_data`
   - Akce: funkce po načtení dat zobrazí jednoduchou tabulku s údaji o uživateli
   - Výstup: vypsání atributů na obrazovku

2. Správa dotazníku

   2.1 Vytvoření dotazníku

   - Vstup: `id_user` právě přihlášeného uživatele
   - Akce: vytvoří se nová instance modelu `Survey` a vykreslí se formulář. Po odeslání formuláře jsou data zvalidována a v případě úspěchu uložena včetně `id_user`, abychom věděli, kdo dotazník vytvořil.
   - Výstup: uložení dat do tabulky `survey`

   2.2 Přidání otázky

   - Vstup: `id_survey` ID dotazníku
   - Akce: vytvoří se nová instance modelu `Question` a vykreslí se formulář. Po odeslání formuláře jsou data zvalidována a v případě úspěchu uložena včetně `id_survey`, abychom věděli, ke kterému dotazníku otázka patří.
   - Výstup: uložení dat do tabulky `question`

   2.3 Přidání volby do otázky

   - Vstup: `id_question` ID otázky

- Akce: vytvoří se nová instance modelu `Choice` a vykreslí se formulář. Po odeslání formuláře jsou data zvalidována a v případě úspěchu uložena včetně `id_question`, abychom věděli, ke které otázce přidaná volby patří.
- Výstup: uložení dat do tabulky `choice`

2.4 Publikování dotazníku

- Vstup: `id_survey` právě přihlášeného uživatele
- Akce: po zavolání požadavku uživatele se provede UPDATE na daném dotazníku prostou změnou jednoho atributu. Atribut `is_active` se z 0 nastaví na 1 a tím se dotazník již začne vypisovat na frontendu portálu.
- Výstup: uložení dat do tabulky `survey`

5.3.0.3 Některé SQL příkazy Zde pro ukázku uvedu aspoň jeden ze složitějších SQL dotazů. Konkrétně to bude dotaz ke zjištění výsledku otázky v podobě vrácení pohledu se sloupci ”value”(samotná odpověď), ”count”(počet zvolení odpovědi) a ”percentage”(procentuální vyjádření). Tento dotaz používám ke zjištění výsledku u otázek typu ”zvolte jednu možnost”. Zde je samotný příkaz:

```
SELECT v.id_question, t.value, count(v.id_choice) as Count,
round(count(v.id_choice)/(select count(*) from rel_choice_response
WHERE id_question = :questionID)*100,2) as Percentage
FROM rel_choice_response v
JOIN tab_choice c on v.id_choice = c.id
JOIN tab_choice_text t on t.id_choice = c.id
GROUP BY v.id_choice having v.id_question = :questionID
ORDER BY Count DESC;
```

6 Zhotovení a zprovoznění webového portálu

6.1 Implementace

6.1.1 Přidání modulů - část admin

6.1.1.1 Zachování struktury Při vytvoření aplikace jsem využil struktury XO a samozřejmě hlavní moduly potřebné k provozu administrace, které byly zmíněny v kapitole 4. Nejprve jsem vymazal všechny nepotřebné moduly (např. `comment`). Poté jsem vytvořil základní strukturu pro můj první modul `survey`, který měl na starost správu dotazníků. Tzn. vytvoření složek `controllers`, `models`, `views` a `components`.

6.1.1.2 Installer Do složky `components` jsem přidal soubor `SurveyInstaller.php`, kde je třída pojmenovaná stejně podle názvu souboru a implementuje rozhraní `XOIInstallComponent`. Toto rozhraní definuje metody `createTables()`, `dropTables()` a `insertRows()`. Podle názvů je jasné, že při zavolání instalátoru XO se spustí tyto metody a provedou potřebné SQl dotazy. V metodě `createTables()` jsou "CREATE TABLE ..."příkazy, jeden pro základní tabulku a druhý pro textovou tabulku. Metoda `dropTables()` obsahuje "DROP TABLE ..."příkazy, pro případné odebrání tabulek z databáze. Poslední jmenovaná metoda `insertRows()` je rovněž moc důležitá, protože bez ní by se Vám modul v administraci neobjevil. Tato metoda obsahuje SQL příkazy naplňující tabulky s přístupovými právy pro uživatele na vytvářený modul.

6.1.1.3 Controller Každý modul samozřejmě potřebuje controller. Analogicky nese název `SurveyController.php`. Rovněž obsahuje stejně pojmenovanou třídu, která dědí ze třídy `XOAdminController`. Třída `XOAdminController` již v sobě obsahuje základní akce pro přidání, editaci, smazání či výpis pro jakýkoliv administrační modul. Tudíž stačilo přepsat jednu metodu `getControllerMenuName`, která vrací řetězec, obsahující název modulu v našem případě "admin/survey/survey/index". Slovo "admin"znamená, že se má použít modul `admin`, "survey"označuje rovněž modul a to `survey`, další slovo "survey"označuje náš controller `SurveyController` a poslední slovo "index"označuje výchozí akci `index`, která vypisuje všechny dotazníky pro přihlášeného uživatele.

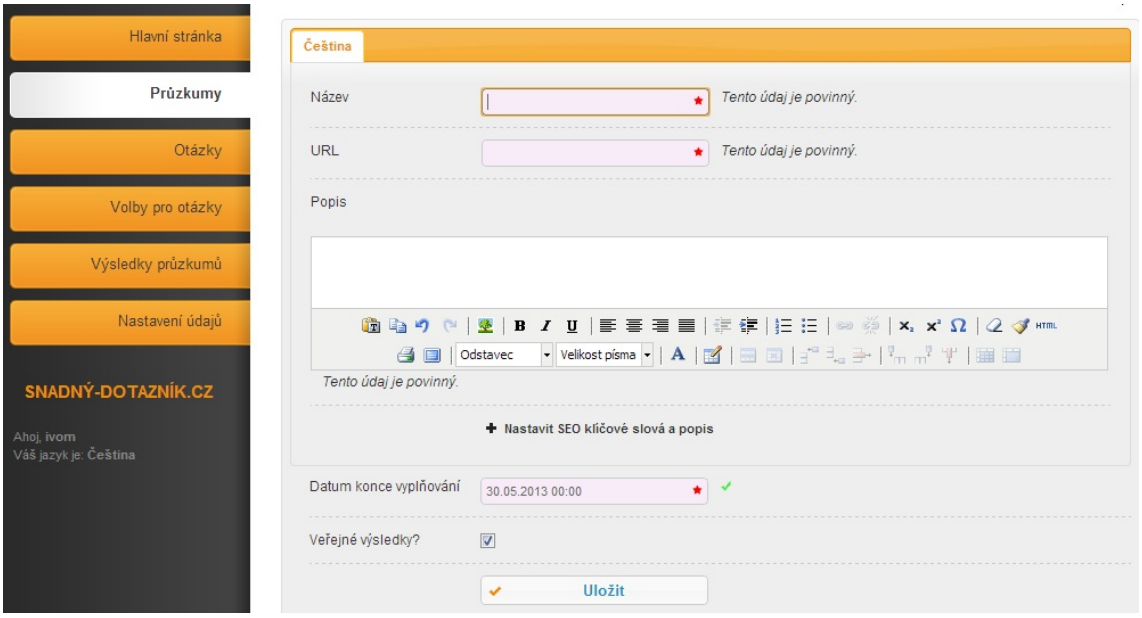
6.1.1.4 Modely Modely jsem musel vytvořit dva, protože dotazník obsahuje atributy jako "title"(název) a "description"(popis). Proto dva modely: `Survey` a `SurveyText`. Model `Survey` dědí ze třídy `XOListModel`, která byla popsána v kapitole 4. Zde bylo potřeba upravit metodu `tableName()`, která vrací řetězec se jménem tabulky. Poté následovala úprava metody `attributeLabels()`, která obsahuje popisky atributů převzatých z tabulky. V poslední řadě jsem upravil metodu `getInternationalDataClassName()` vracející řetězec s názvem textového modelu `SurveyText`, která je použita vždy, když je potřeba vypsat textová data daného objektu. Textový model dědí rovněž třídu `XOListModel`. Zde jsem rovněž přepsal metody `tableName()` a `attributeLabels()`. Poslední přepsanou metodou byla `getMasterForeinKeyName()`, která vrací název atributu, který odkazuje na `id_survey` dotazníku.

Obrázek 4: Hlavní stránka administrace

6.1.1.5 Views Ve složce `views` se vytvoří složka se stejným názvem jako modul, v našem případě `survey`, která obsahuje jednotlivé "view"soubory. Soubory jsou následující `_form.php`, `create.php`, `index.php`, `update.php`, `view.php`. Soubor `index` obsahuje tabulkový výpis všech položek, `create` je určen pro vytváření, `update` pro změnu údajů a ostatní operace se provádí přes `index`. `View` slouží k jednoduchému zobrazení záznamu. Poslední `_form` definuje strukturu formuláře, sloužící jak pro create, tak i update.

6.1.1.6 Ostatní moduly Další moduly potřebné pro provoz mé aplikace jsem vytvořil analogicky podle vytvořeného prvního modulu. Po vytvoření struktur jsem musel moduly ještě připsat do konfiguračního souboru, aby byly viditelné pro aplikaci i pro instalátor. Do sekce (modules => admin => modules =>) jsem připsal názvy modulů, jež jsem vytvořil. A to konkrétně `survey`, `question`, `choice` a `response`. Pak už jen stačilo zavolat instalátor zadaním adresy do prohlížeče např.
`http://localhost/questionnaire/instalace/survey` kde `localhost` je název serveru a `questionnaire` je název mojí aplikace.

6.1.1.7 Uživatelské rozhraní Na obrázku 4 můžete vidět hlavní stránku administrace. Nalevo je umístěné navigační menu, pomocí kterého spravujeme jednotlivé moduly. Nahoře se nachází drobečková navigace. V pravém horním rohu jsou pak umístěna dvě tlačítka. To vlevo slouží k zobrazení webových stránek (frontend) v novém okně a to druhé slouží k odhlášení z administrace. Na obrázku 5 je vyobrazen formulář (konkrétně přidání nového dotazníku). Zde můžete vidět validaci formuláře před odesláním. Po



Obrázek 5: Ukázka formuláře

kliknutí na "Nastavit SEO ..."vyjedou dvě textová pole, kde můžeme zadat SEO popis a klíčové slova sloužící pro lepší vyhledání vytvářeného dotazníku na internetu.

6.1.2 Přidání modulů - část frontend

6.1.2.1 Zachování struktury V případě frontendu jsem při vytváření modulů zachoval úplně stejnou strukturu jako v administrační části. Rozdíly se tu samozřejmě najdou. Za prvé jsem musel při pojmenovávání modulů přidat na začátek malé písmeno "f", které označuje moduly na frontendu. Samozřejmě controller a modely dědí z jiných tříd, než jako v administrační části. Zde jsem vytvořil dva moduly a to `fSurvey` a `fResponse`.

6.1.2.2 Controller Controller `FSurveyController` na frontendu dědí z třídy `XOFrontendController`, která na rozdíl od třídy `XOAdminController` neobsahuje základní zabudované akce jako např. `index` a `detail`. Je to logické, protože každý modul vypisuje své modely jiným způsobem, který je přizpůsoben typu objektu. Já ve svých modulech používám tabulkový výpis.

6.1.2.3 Modely Ve frontendu jsou modely svým obsahem totožné jako v administrační části, ale zde modely nedědí ze třídy `XOListModel`, ale z `XOInterModel`. Rovněž zde používám skoro veškeré modely pro práci s dotazníkem v jednom modulu, na rozdíl od administrační části, kdy jeden model (i se svým textovým) představoval zároveň modul. Toto doporučení mi bylo doporučeno pro jednodušší orientaci, nicméně v budoucnu ve

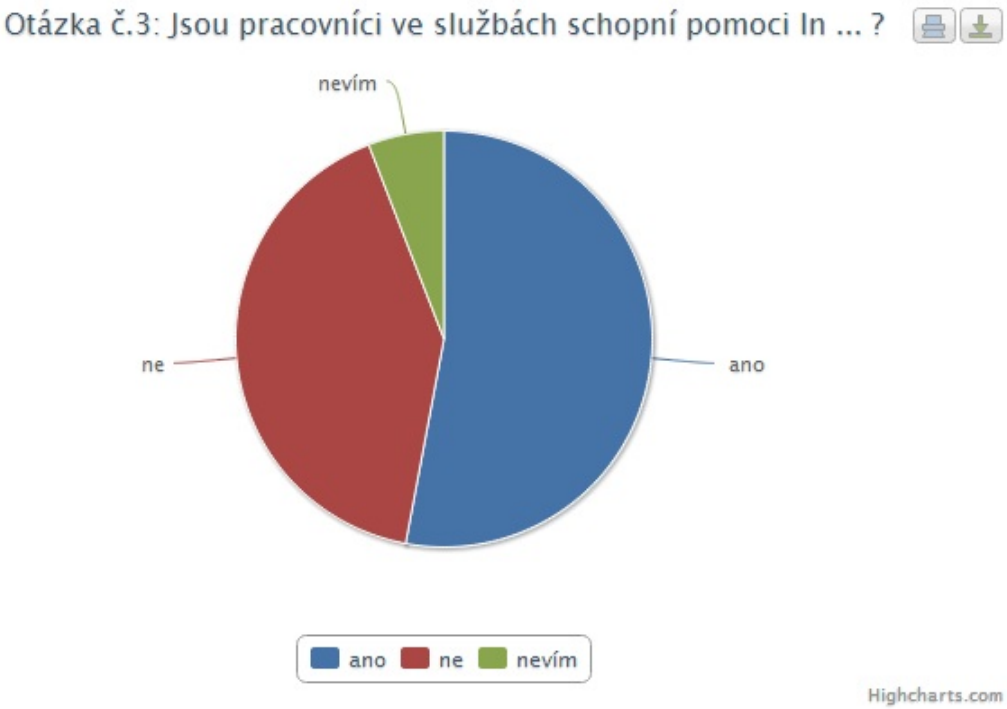
Obrázek 6: Vzhled webového portálu

svých projektech bych se snažil o sloučení blízkých modelů do jednoho modulu jako ve frontendu od XO.

6.1.2.4 Views Zde je podobná situace jako v administrační části, každý soubor představuje jednu akci, ale chybí zde `_form`, který jak víme slouží jako šablona pro formulář, a zde není potřeba.

6.1.2.5 Modul `fSurvey` V tomto modulu jsou dvě standartní akce, a to `index`, která vypíše všechny dostupné dotazníky v tabulkovém výpisu a akce `detail`, která vypíše atributy dotazníku a uživatele, který dotazník vytvořil. Dále je v detailu pod popisem tlačítko sloužící k zahájení vyplňování dotazníku. Stisknutím tlačítka se zavolá akce `fill` (vyplnění dotazníku). Tato akce si kromě načtení modelu dotazníku také načte do listu modely otázek, a pokud otázka má na výběr i možnost volby, tak se pro každou otázku rovněž načte list možností odpovědi. Po vyplnění všech otázek a potvrzení tlačítkem se provede validace, zda-li uživatel vyplnil všechny povinné otázky. Pokud ne, je zobrazen dotazník s jeho odpověďmi, pokud ano, jsou odpovědi zaznamenány a uloženy do databáze.

6.1.2.6 Uložení odpovědí je postupné ukládání modelů vytvořených pro každou otázku, na kterou respondent odpověděl. Nejprve se vytvoří instance modelu `TabResponse`, kde se vytvoří unikátní ID responendta, které se pak používá u každé otázky a model se uloží. Zde se uloží ID respondenta spolu s datumem vyplnění a časem (v sekundách), po který respondent dotazník vyplňoval. Poté se prochází každá otázka a podle toho jakého je typu, se vytvoří nový model, který je poté ihned uložen do databáze.



Obrázek 7: Ukázka grafu HighCharts

6.1.2.7 Modul `fResponse` Tento modul slouží k výpisu již skončených dotazníků a vypisuje jejich výsledky. Vypsány jsou pouze ty dotazníky, u kterých zadavatel zadal, že výsledky jsou dostupné veřejnosti. Akce `index` vypisuje dotazníky v tabulce. Akce `detail` nám vypisuje výsledky každé otázky u dotazníku. Na konci kapitoly 5 je ukázka SQL příkazu, který výsledek otázky zpracovává a vrací v podobě pohledu, jenž je zobrazen v tabulce. Každá tabulka má sloupce `odpověď`, `počet` a `procenta` a pod tabulkou je zobrazen i interaktivní graf, který data znázorňuje v grafické podobě. Pro jejich použití jsem využil doplňku `highcharts`, který po předání dat a nastavení, grafy vykresluje.

6.1.2.8 Uživatelské rozhraní Webové stránky používají jednoduchý styl, který můžete vidět na obrázku 6. V hlavičce je umístěný masivní název webového portálu, který se po přejetí myši plynule zvětšuje a zmenšuje. Pod ním je umístěno navigační menu. Následuje samotný výpis obsahu aktuální stránky. Zde například můžete vidět tabulkový výpis vyhodnocených dotazníků.

6.1.2.9 Doplněk `highcharts` Tento doplněk je napsaný v jazyce JavaScript. Jeho hlavní využití je vykreslování grafů. Ukázku grafu můžete vidět na obrázku 7. Doplněk, který je nastaven pro použití ve frameworku Yii, je volně stažitelný z této adresy: `http://www.yiiframework.com/extension/highcharts/`. Po stažení se doplněk rozbalí do složky `extensions`, která je umístěna v hlavní složce naší webové aplikace.

Obrázek 8: Ukázka třídění položek

Tím máme doplněk připraven a můžeme ho začít používat. V ukázce 3 je kód, který doplněk spustí. Na prvním řádku je klíčové slovo `Widget`, které framework pozná a zinicializuje doplněk pomocí prvního parametru na druhém řádku, kde je cesta ke spuštění doplňku. Na posledním řádku je parametr, který obsahuje asociativní pole, které je vráceno metodou v controlleru.

```
$this->Widget(
    'ext.highcharts.HighchartsWidget',
        $this->getChartPieOptions($chart_title, $chart_data));
```

Výpis 3: Ukázka práce s widgetem

6.2 Řešení problému třídění dat

6.2.0.10 Akce Sort Tato akce slouží k interaktivnímu třídění položek. Na obrázku 8 je ukázka, jak vypadá uživatelské prostředí pro třídění položek. Při této akci jsem narazil na problém, jak zobrazit výpis otázek pouze právě přihlášeného uživatele, který je rovněž zadavatelem a vlastníkem dotazníku. Ve výpisu 4 je ukázka kódu, který řeší správné nastavení a předání dat.

```
$listSurvey = array();
$modelSurvey = Survey::model()->findAll('is_active=:active', array(':active' => 0));
foreach ($modelSurvey as $survey) {
    array_push($listSurvey, $survey->id);
}

// soubor sort.php

$url = Yii::app()->createUrl("admin/question/question/sort");
XOAdminViewHelpers::openSortableData($url);
```

```
$position = 1;
foreach( $models as $model ) {
  if (in_array($model->id_survey, $listSurvey) and $this->surveyOwns($model->id_survey)) {
    $lModel = $model->getInternationalModel(XOUtilities::getCurrentUserLanguageId());
     $title = $lModel->title;
    $survey = Survey::model()->find('id=:id', array(':id' => $model->id_survey));
    $surveyText = $survey->getInternationalModel(XOUtilities::getCurrentUserLanguageId());
    $surveyTitle = $surveyText->title;
     if (strlen($surveyTitle) > 50) { $surveyTitle = substr($surveyTitle, 0, 50) . " ... "; }
     if (strlen( $title ) > 50) { $title = substr( $title , 0, 50) . " ... ?"; }
    $text = $position . ". " . $title . " (Pruzkum:" . $surveyTitle . ")";
    XOAdminViewHelpers::bodySortableData($model->id, $text);
    $position++;
  }
}
XOAdminViewHelpers::closeSortableData();
```

Výpis 4: Hlavní kód akce Sort

6.2.0.11 Popis řešeného kódu První blok kódu má za úkol vytvořit pole obsahující ID dotazníků, které mají nastavený atribut `is_active` na hodnotu `0` (tzn. že dotazník ještě nebyl publikován). Nejprve si načtu seznam modelů (`survey`) a poté procházím seznam a ukládám jednotlivá ID do pole `$listSurvey`. Tento blok je umístěn v controlleru a společně se seznamem otázek je předán souboru `sort.php`. Další dva řádky slouží k inicializaci tabulky pro třídění.

6.2.0.12 Přidání dat do tabulky Poté následuje cyklus *foreach*. S jeho pomocí procházím jednotlivé modely otázek předané z controlleru (`$models`). Následuje podmínka, která ověřuje, zda-li je `id_survey` modelu otázky obsažen v poli dotazníků vytvořeném výše. Rovněž podmínka kontroluje, zda-li otázka odkazuje na dotazník, jenž je jeho vlastníkem právě přihlášený uživatel (metoda `surveyOwns()`). Následuje získání titulku otázky, nejprve načteme textový model a poté přiřadíme atribut "title". Další tři řádky slouží k získání titulku dotazníku. Nejprve načteme model dotazníku na zákadě předání `id_survey`, poté načtení textového modelu a přiřazení titulku. Hned za získáním obou titulků jsou umístěny podmínky pro ořezaní titulků, pokud jsou delší než 50 znaků. Následuje zformátování obou titulků s výpisem čísla pořadí do proměnné `$text`. Posléze je otázka přidána do výpisu tabulky předáním její ID a vytvořeného a zformátovaného textu. Za cyklem je ukončení sběru dat tabulky.

6.3 Zprovoznění a spuštění portálu

6.3.1 Instalace portálu

6.3.1.1 Potřebné vybavení Pro provoz portálu samozřejmě budeme potřebovat framework Yii. Dále budeme potřebovat webový server, který podporuje PHP verzi 5.1.0

nebo vyšší a databázový server MySQL verze 5.5.x nebo vyšší. Pokud již na svém operačním systému servery nainstalované máte, můžete následující odstavec přeskočit.

6.3.1.2 Instalace serverů Pro jednodušší práci doporučuji stáhnout aplikační balík, který v sobě obsahuje jak webový server, tak databázový, včetně webové administrace databází. Mně se nejvíce osvědčil balík "Ampps", který je navíc i v češtině. Je volně stažitelný z této adresy `http://www.ampps.com/downloads`. Po instalaci zapneme a jsme připraveni na další krok.

6.3.1.3 Framework Yii Framework je dostupný na této adrese . Link na stažení je uprostřed stránky vedle pole "Source Code", zabalený v archívu. Archív si rozbalíme a z něho zkopírujeme složku `framework` do složky našeho webového serveru. Pokud používáte platformu "Windows"a balíček "Ampps", je cesta ke složce webového serveru `C:/Program Files/Ampps/www`.

6.3.1.4 Webová aplikace Do složky webového serveru rovněž zkopírujeme webovou aplikaci, která je umístěna na přiloženém CD. Z CD si stáhneme a rozbalíme archív `prilohy.zip` a jeho obsah zkopírujeme rovněž do webové složky serveru, hned vedle složky `framework`. Nnyí je ještě potřeba nastavit připojení k databázi.

6.3.1.5 Databáze Posledním krokem ke správnému chodu portálu je přidání databáze a naplnění ji daty. Nejdříve si spustíme webového klienta phpmyadmin zadáním adresy `http://localhost/phpmyadmin`. Nahoře v navigačním menu zvolíme "Databáze". Poté zadáme název nové databáze "questionnaire"(pokud bychom zadali jiný název, museli bychom v konfiguračním souboru název databáze přepsat). Zvolíme kódování z roletky "utf8_unicode_ci"a potvrdíme tlačítkem "Vytvořit". V levém panelu klikneme na naši nově vytvořenou databázi. Nyní z horního menu vybereme "Import". Klikneme na "vybrat soubor"a v archivu `prilohy.zip` umístěném na CD je soubor `databaze.sql`. Tento soubor vybereme a potvrdíme dole talčítkem "Proveď". Soubor se zpracuje a nahoře se nám vypíše hláška o úspěšném importu.

6.3.2 Spuštění portálu

6.3.2.1 Zobrazení stránek Pokud jsme dodrželi všechny kroky instalace, měla by se nám zobrazit úvodní stránka po zadání adresy `http://localhost/questionnaire/`. Stránky by měly být funkční a měli by obsahovat dva již vyhodnocené dotazníky dostupné na adrese: `http://localhost/questionnaire/response/vypis/`. Pro přihlášení do administrační části použijte navigační menu "Přihlásit se"nebo zadejte adresu `http://localhost/questionnaire/admin/`. Zobrazí se přihlašovací obrazovka ve žlutooranžovém panelu. Pro přihlášení použijte login "xoadmin"a heslo je také "xoadmin". Po úspěšném přihlášení se zobrazí hlavní stránka s popisem jak administraci ovládat a vyznat se v ní.

6.3.2.2 Online verze K dispozici je i verze umístěná na internetu. Je dostupná na této adrese `http://snadny-dotaznik.xostudio.cz/`. Pro přihlášení do administrace lze použít jako login "host"a heslo rovněž "host". Zde je možné si vyzkoušet kompletní správu dotazníků včetně jejich vyhodnocování.

6.3.2.3 Problém se spojením databáze Pokud používáte svůj systém, na který jste zvyklý a máte nastaveného uživatele "mysql"pro přístup do databáze bez hesla či s jiným heslem, je potřeba toto nastavit v konfiguračním souboru. V hlavní složce projektu je `protected` a v ní složka `config` a v ní soubor `main.php`. Ten otevřete ve svém oblíbeném textovém editoru. Najděte část "connectionString". Pokud byl použit jiný název databáze, musí se za "dbname="dopsat název. Pro ověření uživatele změňte "username"a "password", na který jste zvyklý a používáte jej.

7 Závěr

V první části bakalářské práce byl vypracován popis webového frameworku Yii. Byly zde popsány základní vlastnosti frameworku, jeho používání a využití pro webové aplikace. Dále byla popsána vrstva XO dodána firmou XO Studio, která přináší zjednodušený a rychlý vývoj webových aplikací. Dále zavádí jisté standardy pro vývoj a spravování webových aplikací.

Ve druhé části byl proveden návrh webového portálu a jeho realizace. Portál měl být určen pro správu dotazníků a jejich vyhodnocování a dále měl využívat framework Yii a vrstvu XO. Portál se mi podařilo vytvořit a zprovoznit do podoby, která byla požadována zadáním bakalářské práce.

Pokud mám zhodnotit část frontendu, přijde mi návrh a realizace vyřešeny velice elegantně a jednoduše. Všechny tabulkové výpisy jsou přehledné a mají jednotný styl. Určitě potěší i znázornění výsledků pomocí grafů. Administrační část se zpočátku může zdát trochu nepřehledná. Proto jsem na hlavní stránce umístil interaktivní menší manuál, jak se v administraci orientovat a ovládat. Všechny formuláře obsahují validaci a jsou velice jednoduché a přehledné.

Projekt se samozřejmě dá nadále rozšiřovat, přidávat nové typy otázek či přidat stáhnutí vyhodnocení dotazníku v podobě souboru `.pdf`, `.xls` a dalších. Pro ukázku bych mohl jmenovat portál "Vyplňto.cz", který podle mého názoru zvládá problematiku online dotazníků na jedničku.

Tato bakalářská práce mi přinesla hodně nového do mého života. Především je to framework Yii, který jsem shledal jako velice výkonný nástroj pro vývoj webových aplikací a už nyní své další projekty stavím na tomto frameworku. Je jednoduchý, má pěkně zpracovanou dokumentaci a neustále se zvětšující komunitu.

Ivo Michalík

8 Reference

[1] *Design Patterns and Refactoring* [online]. c2009. [cit. 2013-04-02]. Dostupné z: <http://sourcemaking.com/návrhový-vzor>.

[2] *WiseGEEK: clear answers for common questions* [online]. c2003. [cit. 2013-04-02]. Dostupný z: <http://www.wisegeek.com/what-is-a-web-application-framework.htm>.

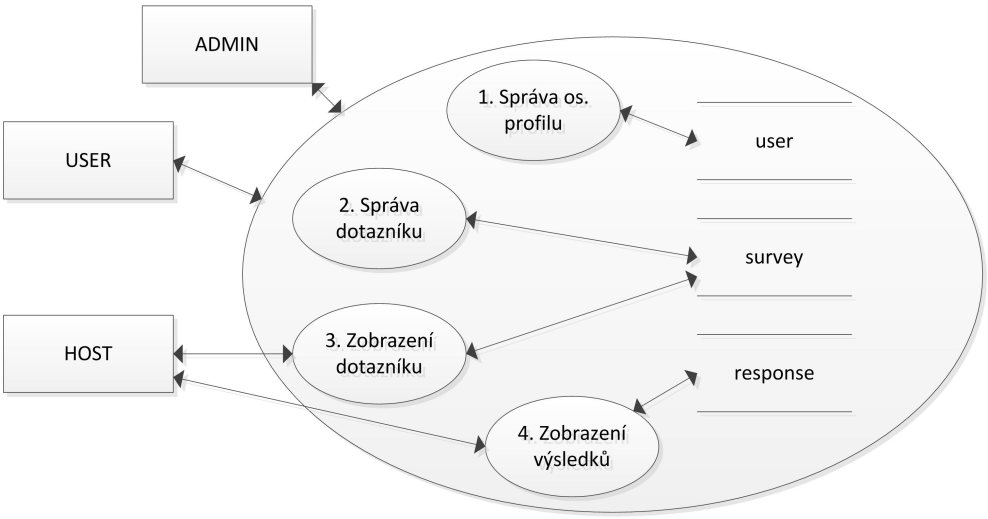
[3] *Zdroják — o tvorbě webových stránek a aplikací* [online]. c2009. [cit. 2013-04-04]. Dostupný z: <http://www.zdrojak.cz/clanky/uvod-do-architektury-mvc/>.

[4] *JSON* [online]. c1999. [cit. 2013-04-08]. Dostupný z: <http://www.json.org/json-cz.html>.

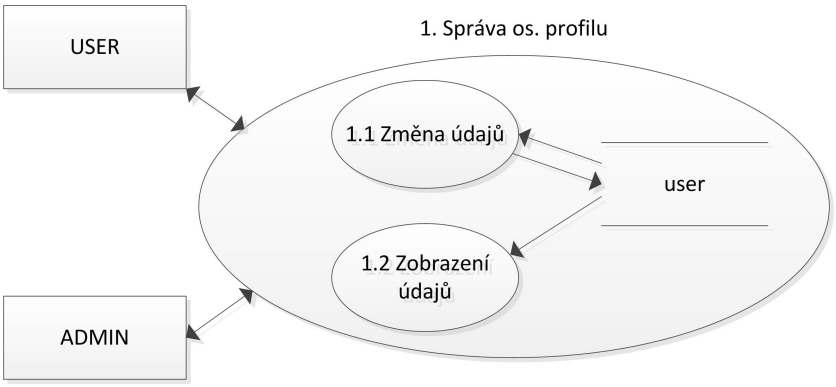
[5] WINESETT, Jeffrey. *Agile Web Application Development with Yii 1.1 and PHP5*, Birmingham: Packt Publishing Ltd., 2010

[6] *Fundamentals: Model-View-Controller (MVC)* [online]. c2013. [cit. 2013-04-14]. Dostupný z: <http://www.yiiframework.com/doc/guide/1.1/en/basics.mvc>.

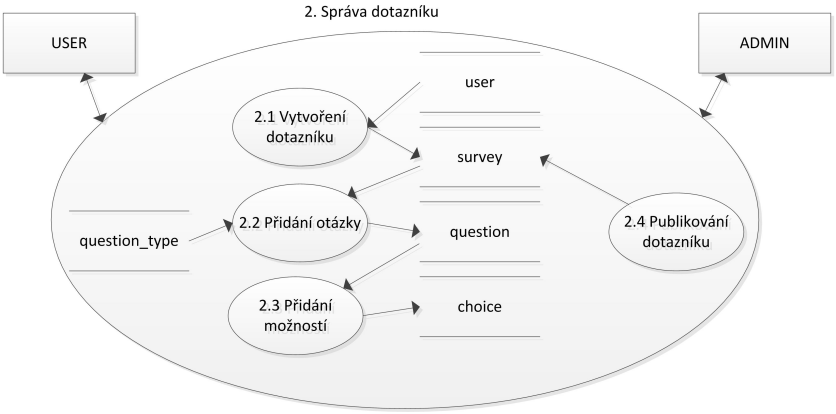
A Obrázky



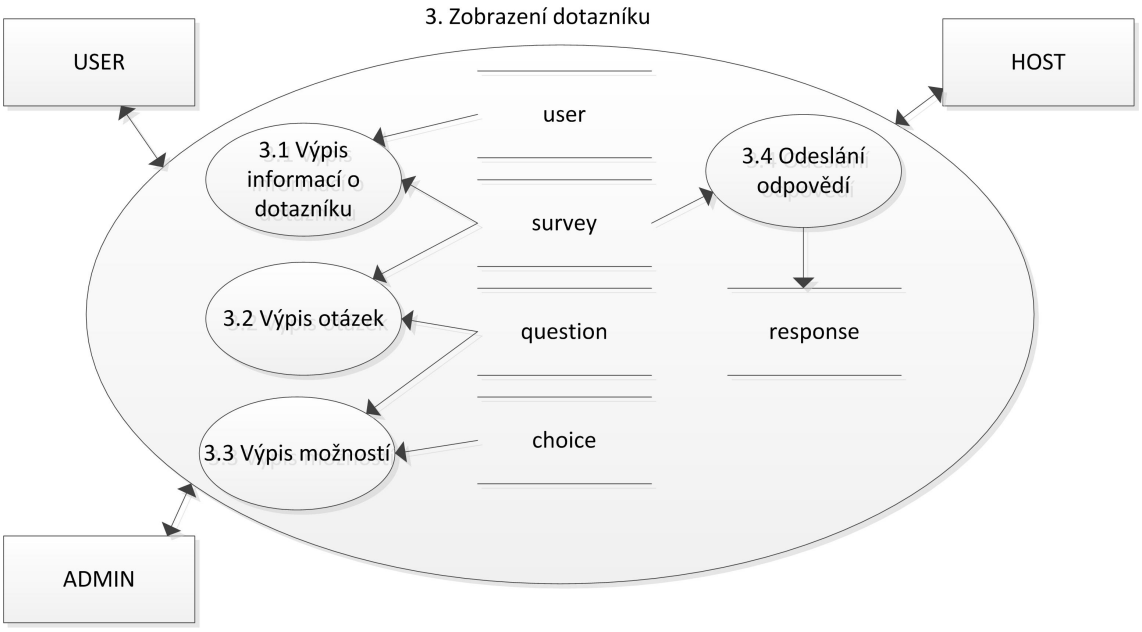
Obrázek 9: Diagram DFD úrovně 0



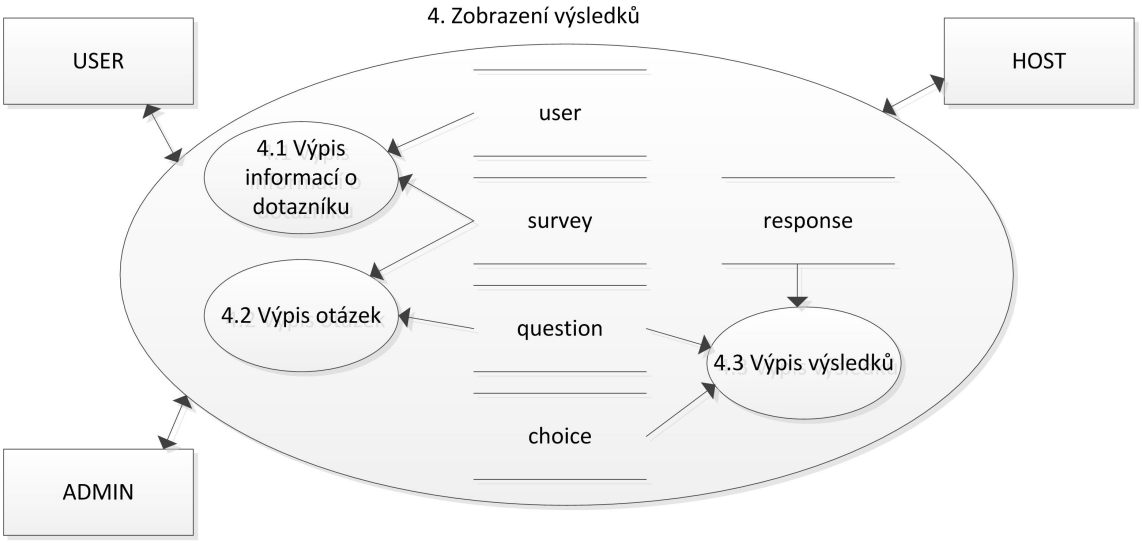
Obrázek 10: Diagram DFD úrovně 1



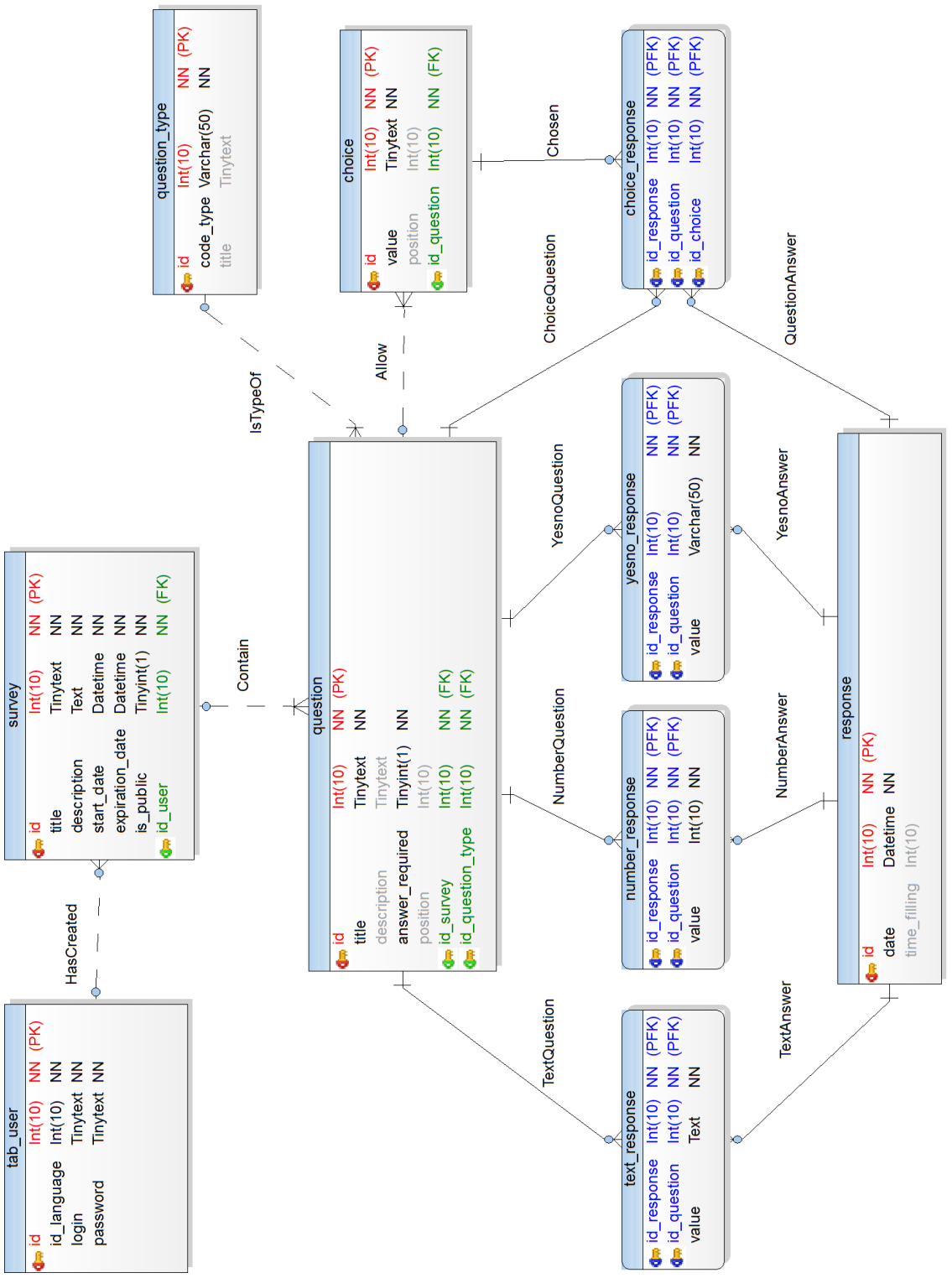
Obrázek 11: Diagram DFD úrovně 1



Obrázek 12: Diagram DFD úrovně 1



Obrázek 13: Diagram DFD úrovně 1



Obrázek 14: Databázový model

B Tabulky

Tabulka	Atribut	Datový typ	Velikost	Klíč	Null
user	id	int	10	PK	
user	id_language	int	10		
user	login	tinytext			
user	password	tinytext			
survey	id	int	10	PK	
survey	title	tinytext			
survey	description	text			
survey	start_date	datetime			
survey	expiration_date	datetime			
survey	is_public	tinyint	1		
survey	id_user	int	10	FK1	
question	id	int	10	PK	
question	title	tinytext			
question	description	text			Null
question	answer_required	tinyint	1		
question	position	int	10		Null
question	id_survey	int	10	FK1	
question	id_question_type	int	10	FK2	
question_type	id	int	10	PK	
question_type	code_type	varchar	50		
question_type	title	tinytext			
choice	id	int	10	PK	
choice	value	tinytext			
choice	position	int	10		Null
choice	id_question	int	10	FK1	
response	id	int	10	PK	
response	date	datetime			
response	time_filling	int	10		Null
choice_response	id_response	int	10	PK, FK1	
choice_response	id_question	int	10	PK, FK2	
choice_response	id_choice	int	10	PK, FK3	
text_response	id_response	int	10	PK, FK1	
text_response	id_question	int	10	PK, FK2	
text_response	id_choice	int	10	PK, FK3	
number_response	id_response	int	10	PK, FK1	
number_response	id_question	int	10	PK, FK2	
number_response	id_choice	int	10	PK, FK3	
yesno_response	id_response	int	10	PK, FK1	
yesno_response	id_question	int	10	PK, FK2	
yesno_response	id_choice	int	10	PK, FK3	

Tabulka 1: Tabulka s popisem atributů